UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/544,512	04/06/2000	Corneliu I. Lupu	MSFT114614	9057

26389 7590 11/24/2003

CHRISTENSEN, O'CONNOR, JOHNSON, KINDNESS, PLLC
1420 FIFTH AVENUE
SUITE 2800
SEATTLE, WA 98101-2347

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2124	7

DATE MAILED: 11/24/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PRG

Office Action Summary	Application No.	Applicant(s)	
	09/544,512	LUPU ET AL.	
	Examiner	Art Unit	
	Tuan A Vu	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 September 2003* .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-18* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-18* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *06 April 2000* is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

DETAILED ACTION

1. This action is responsive to the Applicant's response filed September 8, 2003.

Claims 2-18 have been amended. Claims 1-18 are pending in the office action.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1, 3-4, 7, 9, 13, 15, and 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Straub, USPN: 5,430,878 (hereinafter Straub), in view of Nowlin, Jr. et al., USPN: 6,484,309 (hereinafter Nowlin)

As per claim 1, Straub discloses a method for patching a computer application program comprising:

determining whether or not the computer application program is compatible with a computer operating system executing the application (e.g. col. 4, line 4 to col. 5, line 27); and

if the computer application program is determined to be incompatible with the computer operating system, starting a code replacement process to address the incompatible matching(e.g. step *110* -Fig. 1; steps *302, 305* – Fig. 3).

But Straub does not explicitly specify starting up a debugger upon determining an incompatibility with the operating system, although teaches invoking special programs to revise code and to insert instructions to patch the non-compatible sections of code under revision (e.g.

col. 5, line 28 to col. 6, line 25). In a method to correct application code executing under a different operating system using memory scanning and code replacing analogous to the compatibility code patching by Straub, Nowlin discloses calling upon a combination of surrogate code and collaboration set of procedures and filter DLL to perform the O.S. conversion to address the compatibility issue, hence suggested invoking calls to a debugger process (e.g. col. 62 to col. 3, line 39; Fig. 2-3; col. 4, line 34 to col. 5, line 42). It would have been obvious for one of ordinary skill in the art at the time the invention was made to apply a combination of code for debugging the O.S. incompatibility issue when running application as taught by Nowlin and add this process to Straub's method of correcting for O.S. compatibility because according to Nowlin, this would provide smooth and transparent transitions of a wide variety of applications into a more compatible O.S. format without having to recompile or rewrite code thereby providing a more efficient and/or universal way of translating Window based applications for different platforms (Nowlin: col. 1, lines 27-52).

As per claim 3, Straub further discloses

determining if at least one identifying attribute of a plurality of identifying attributes of the computer application program does not match at least one identifying attribute of a plurality of identifying attributes of compatible applications (e.g. *version numbers*—col. 4, lines 32-54;); and

if at least one of the identifying attributes matches, determining if the computer application program is incompatible, otherwise it is compatible (e.g. col. 5, lines 1-27).

But Straub does not specify matching identifying attributes against attributes of incompatible applications. But in view of the disclosed matching against known patterns by

Straub, one of ordinary skill in the art would recognize that the techniques by Straub would have achieved the same results as the claimed limitation suggests; and would also be motivated to modify such pattern matching by comparing such attributes against known incompatible application attributes in case the availability of such incompatible applications is handy for use without additional resources spending because it would have been obvious that providing not only the known compliant but also the known non-compliant patterns, i.e. incompatible attributes as claimed, would make the incompatibility checking even more adequate.

As per claim 4, Straub further discloses storing identifying attributes of compatible applications (e.g. *store table 201* – Fig. 2); and retrieving one of such stored identifying attributes for determining one of the attributes of the computer application program matches the stored attributes of the compatible applications (e.g. col. 5, lines 1-27).

But Straub does not specify storing and using attributes of incompatible applications for the matching against attributes of the target application; but this limitation would have been obvious by virtue of the rationale set forth in claim 3 above.

As per claim 7, this is the computer-readable medium version of claim 1 above, hence incorporates the rejection thereof for the same obvious reasons; and further includes a computer-readable medium to embody the debugging method, which Straub does not disclose. Official notice is taken that the use of a computer-readable medium to store a computer product program code was a well-known concept at the time of the invention. Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to use a computer-readable medium to store the computer product program code as disclosed by Straub because this would

facilitate the distribution/sale of such computer product and the use of such product by a broader population of computer users.

As per claim 9, this is the computer-readable medium version of claim 3, hence incorporates the corresponding rejections set forth in therein for the same reasons.

As per claim 13, this is the system version of claim 1 above, hence incorporates the rejection thereof for the same obvious reasons.

As per claim 15, in reference to claim 13, this is the computer system version of claim 3, hence incorporates the corresponding rejections set forth in therein for the same reasons.

As per claim 16, this is the system version of claim 4, hence incorporates the corresponding rejections set forth in therein for the same reasons.

4. Claims 2, 5-6, 8, 10, 11-12, 14, 17 and 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over over Straub, USPN: 5,430,878, in view of Nowlin, Jr. et al., USPN: 6,484,309, as applied to claims 1, 7, 13, 15 in view of Preisler et al., USPN: 5,675,803 (hereinafter Preisler).

As per claim 2, Straub discloses specific area of image memory at which the revising process stops and applies patches (e.g. col. 3, line 66 to col. 4, line 31; Fig. 2-3 – Note: interrupting a revision process to insert code is equivalent to setting breakpoint for patching); and Nowlin teaches executing the code reconversion in a debugger with code relocation and entry points information without having to recompile (e.g. col. 6, line 40 to col. 7, line 34).

But Straub (with Nowlin's teaching) does not specify setting of breakpoints so to monitor the application via the debugger to determine if at least one such breakpoint has been reached; and applying the patching thereupon; and executing the steps of running the application, monitoring, patching until the application has finished. In the same approach for patching

without compiling as suggested by Nowlin, Preisler, in a method to run a "Fix-and-Continue' type of debug to patch target application while checking for run-time errors analogous to the dynamic application patching process by Straub/Nowlin, discloses setting and monitoring for patch sites, i.e. breakpoints while running the application (*patch sites* -- col. 61-65), and applying the patching process upon such breakpoints (*patch site 20 → Load Instruction*, Fig. 3) and re-executing the steps of monitoring and patching until the application has finished (steps 110, 130, 140, 160 - Fig. 2; col. 6, lines 1-64). It would have been obvious for one of ordinary skill in the art at the time the invention was made to arrange Straub's debugging process so that pre-defined breakpoints so suggested by Preisler are inserted in the process of using patching or filtering as taught by Straub (and modified by Nowlin), and monitor for such breakpoints during the execution of the application because this would further enhance control over the dynamic state of change of the application program and thereby assert more correctness checking using pre-defined points at which closer data recording or execution code patching as mentioned by Straub/Nowlin or could help debug or apply fix to the application in a more controllable and predictable way.

As per claim 5, Straub in combination with Preisler discloses executing a debugger containing a set or list of breakpoints, each breakpoint having a set of instructions for patching the application (see claim 2); the debugger setting a set breakpoints within the application (see claim 2). Further, Nowlin teaches invoking a filter code used in conjunction with other collaboration DLL (Figs 2-3) for debugging the application under execution in an heterogenous O.S., such code being a dynamic linked library (e.g. col. 4, lines 34-41).

But Straub (with Nowlin teachings) fails to disclose accessing the list of breakpoints from such debugger DLL to set breakpoints. However, Preisler discloses setting of patch points while running the debugger (*patch sites* -- cols. 61-65) and Straub suggests stopping points for the patching process to apply replacement code from the scanning process (e.g. Fig. 3), hence in combination they suggest the teaching as to accessing the list of breakpoints from such debugger. In view of the general knowledge that dynamic linked libraries as used by Nowlin can be compiled externally and used selectively in certain platforms while protected against dynamic modification, it would have been obvious for one of ordinary skill in the art at the time the invention was made to create a debugger as a DLL as suggested by Nowlin and apply this form of DLL debugger to the suggested breakpoint patching techniques of Straub/Preisler and thereby access a list of pre-set breakpoints as suggested above by Straub/Preisler in the intent to implement a breakpoint/patching technique in the debugger. One of ordinary skill in the art would be motivated to do so because, as suggested by the general knowledge that DLL is a form of executable that can be loaded in compatible platform, resistive to changes, and occupying small storage places but highly reusable because of their properties to be linked and dynamically indirectly invoked or wrapped at runtime of a given application, hence facilitate the debugging process and breakpoint presetting of Straub/Preisler in certain operating platforms in which compliance for execution is to be respected; and conformity to such O.S. is what Straub invention is all about.

As per claim 6, Straub (with Preisler's teaching) further discloses upon reaching a breakpoint where compatibility check fails, calling code replacement to patch area instructions set (e.g. Fig. 3), and patching the incompatible application based on such instructions. Straub

does not explicitly specify calling a handler associated with a breakpoint; but in view of the use of debugger DLL by Nowlin, this limitation would have been obvious for the same rationale as set forth in claim 1 and/or claim 5 above.

As per claim 8, this is the computer-readable medium version of claim 2; hence incorporates the corresponding rejection set forth therein for the same reasons.

As per claim 10, this is the computer-readable medium version of claim 4, hence incorporates the corresponding rejections set forth in therein for the same reasons.

As per claim 11, this is the computer-readable medium version of claim 5, hence incorporates the corresponding rejections set forth in therein for the same reasons.

As per claim 12, this is the computer-readable medium version of claim 6 above; hence incorporates the corresponding rejections set forth in therein for the same reasons.

As per claim 14, this is the system version of claim 2, hence incorporates the corresponding rejections set forth in therein for the same reasons.

As per claim 17, this is the system version of claim 5, hence incorporates the corresponding rejections set forth in therein for the same reasons.

As per claim 18, refer to claim 6 for the same rationale.

Conclusion

5. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Pat No. 6,138,271 to Keeley, disclosing scanning code for operation in different OS kernel and conversion.

U.S. Pat No. 5,473,777 to Moeller et al., disclosing OS based conversion of procedural native code into a OO program.

U.S. Pat No. 6,483,583 to Hammond, disclosing injection of code in kernel DLL to match OS conformity.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 872-9306 (for formal communications intended for entry)

or: (703) 746-8734 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA. , 22202. 4th Floor(Receptionist).

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

VAT
October 28, 2003

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100